MLPR – 2023 – Fingerprint spoofing detection

The goal of the project is the development of a classifier to detect whether a fingerprint image represents an authentic fingerprint or a spoofed fingerprint, i.e. a fingerprint image obtained by cloning, possibly with different methods, the fingerprint of an individual. The fingerprint images are represented by means of embeddings, i.e. low-dimensional representations of images obtained by mapping images to a low-dimensional manifold (typically few hundred dimensions). To keep the model tractable, the dataset consists of synthetic data, and embeddings have significantly lower dimension than in real use-cases.

The embeddings are 10-dimensional, continuous-valued vectors, belonging to either the *authentic fingerprint* (label 1) or the *spoofed fingerprint* (label 0) class. The embedding components do not have a physical interpretation.

File `Train.txt` contains the embeddings that can be employed to build the classification model (training set). The evaluation embeddings (evaluation set) are provided in file `Test.txt`. Each row of each file correspond to a sample. The features and corresponding label of each sample are separated by commas, with the first 10 columns of each file corresponding to the sample components, whereas the last column contains the corresponding label.

The datasets are imbalanced, with the spoofed fingerprint class having significantly more samples. The spoofed fingerprint samples belong to one of 6 possible sub-classes, corresponding to different spoofing approaches, but the actual spoofing method (sub-class label) is not available. The target application considers an application where prior probabilities of authentic and spoofed classes are the same, but labeling a spoofed fingerprint as authentic has a larger cost due to the security vulnerabilities that such errors would create. The target application working point is threrefore defined by the triplet $(\pi_T = 0.5, C_{fn} = 1, C_{fp} = 10)$.

Model training and model selection

The report should provide an analysis of the dataset and of suitable models for the task, and the methodology employed to select a candidate solution among different possible alternatives (e.g. different classification models, different values of the hyperparameters, different pre-processing strategies).
The models must be trained over the training set only. When needed, validation data can be extracted from the training set (for example, to compare competing models, to select suitable values for the hyperparameters of each model, or to train score calibration models). Models should be trained to optimize the target application, but performance of the models for alternative applications should also be analyzed and discussed. At the end of this stage, a candidate solution should be provided for the classification task.

Evaluation of the candidate model

The proposed solution must be evaluated on the evaluation set. The evaluation samples should be treated as independent, i.e. the value or the score of an evaluation sample should have no influence on the score of a different sample. The evaluation should report the performance in terms of the primary metric, but also analyze how the model would perform for alternative applications (i.e. alternative working points).

Post-evaluation analysis

The choices made during model training should be analyzed to verify that the selected model is indeed competitive for the task. This requires comparing on the evaluation set alternative models that were analyzed and trained, but discarded, in the first phase, to verify whether the proposed solution is indeed optimal or close to optimal with respect to possible alternatives (e.g., the chosen models is effective, the chosen hyperparameters are optimal, etc.).